Constructing Stochastic Mixture Policies for Episodic Multiobjective Reinforcement Learning Tasks

Peter Vamplew, Richard Dazeley, Ewan Barker and Andrei Kelarev

Graduate School of Information Technology and Mathematical Sciences
University of Ballarat, University Drive, Mount Helen, Victoria 3353, Australia
{p.vamplew, r.dazeley, e.barker, a.kelarev@ballarat.edu.au}

Abstract. Multiobjective reinforcement learning algorithms extend reinforcement learning techniques to problems with multiple conflicting objectives. This paper discusses the advantages gained from applying stochastic policies to multiobjective tasks and examines a particular form of stochastic policy known as a mixture policy. Two methods are proposed for deriving mixture policies for episodic multiobjective tasks from deterministic base policies found via scalarised reinforcement learning. It is shown that these approaches are an efficient means of identifying solutions which offer a superior match to the user's preferences than can be achieved by methods based strictly on deterministic policies.

Keywords: multiobjective, reinforcement learning, scalarisation, Pareto fronts

1 Introduction

The vast majority of reinforcement learning (RL) algorithms deal with tasks involving maximising performance on a single objective, as encoded in the scalar reward received from the environment. Whilst many problems can naturally be described by this model, over recent years there has been growing recognition within the optimisation community that many real-world problems require the optimisation of multiple, often conflicting, objectives [1]. This observation is equally true for RL tasks – for example, [2] applied reinforcement learning to simultaneously manage the power consumption and performance of Web application servers. Recently there has been some interest in extending existing single-objective RL methods to handle multiobjective tasks. However one issue which has been largely overlooked in the extension of these approaches is the potential benefits to be gained in the multi-objective domain by moving from deterministic to probabilistic policies.

Section 2 of this paper briefly reviews existing approaches to multiobjective reinforcement learning (MORL). Section 3 discusses the relationship between multiobjective tasks and probabilistic policies, presenting an example to motivate further investigation. Section 4 explores one specific type of probabilistic policy (the mixture policy), and Section 5 investigates means by which mixture policies can be generated from deterministic policies. Finally Section 6 offers suggestions for future directions for research in learning mixture policies for multiobjective tasks.

2. Multiobjective Reinforcement Learning

Before reviewing existing approaches to MORL, it is instructive to examine some fundamental concepts underlying the analysis of multiobjective tasks. Inherently the task in any multiobjective situation is to identify solution(s) which represent a 'good' compromise amongst the objectives. This is often defined via Pareto dominance which allows comparison of a pair of solutions, as shown in Fig 1[1]. A solution dominates another if it is superior on at least one objective, and at least equal on all other objectives. Two solutions are incomparable if each is superior to the other on at least one objective. Any dominated solution is of little value, as clearly the dominating solution is preferable. Therefore the best solutions in a set can be extracted by retaining only those which either dominate or are incomparable with every other member of the set. If this process is applied to the set of all solutions, the resulting set of non-dominated solutions is referred to as the Pareto optimal front (or the Pareto front), and represents the globally optimal set of compromise solutions (see Fig 2). Of course establishing the true front for any problem of significant size is generally impractical, and so the goal of many multiobjective problem-solvers is to produce an approximation to this front. A good approximate front should contain solutions which are accurate (close to the actual front) and well distributed along the front, with a similar extent to that of the actual front.

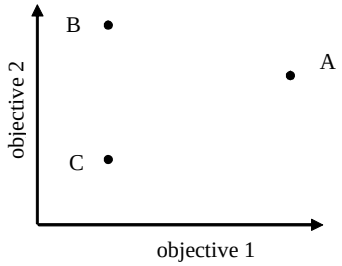

Fig 1: Solutions A and B dominate solution C; solutions A and B are incomparable to each other.

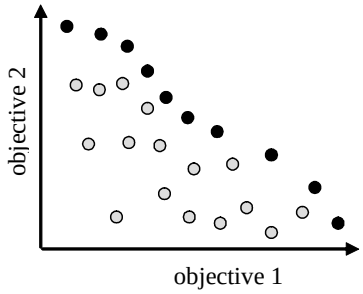

Fig 2: The black points indicate solutions which form the Pareto front; all grey solutions are dominated by at least one member of the Pareto front.

There are several advantages to searching for a set of compromise solutions rather than attempting to find a single 'optimal' solution. Methods which aim for a single solution require *a priori* decisions from the user about the desired nature of that solution (e.g. specifying weights or thresholds for objectives). This requires domain knowledge on the part of the user, and minor variations in these preferences may result in significant variations in the solution achieved, which can easily lead to the

[1] In multiobjective optimisation, the task is generally to minimise each objective, so a lower value for an objective is superior to a higher value. In contrast in RL the task is to maximise the reward received, and so the notion of superiority is reversed. Given the expected audience for this paper, we chose to frame our discussion in terms of maximisation.

acceptance of an inferior solution. For example, a slightly higher threshold for one objective may prevent discovery of a solution which provides a significant improvement on all other objectives. Systems which produce sets of solutions allow *a posteriori* decisions about the solution to be accepted, which are easier and better informed as they are based on knowledge of the trade-offs available as encapsulated by the front. Also the presentation of the front to the user may provide better insight into the relationships between the objectives. The primary disadvantage of generating multiple solutions rather than a single solution is the increased computational cost. In addition in the case of on-line learning in a real environment, the losses incurred during the extended learning period may prohibit searching for the complete Pareto set of policies.

The most straightforward and most common means of extending existing RL algorithms to multiobjective problems is to convert the problems into single-objective tasks. The key difference between single-objective and multiobjective RL is that in the former the reward is scalar, whereas in the latter it is a vector, with an element for each objective. Therefore a multiobjective task can be reduced to a single objective via the process of scalarisation, where a function is applied to the reward vector to produce a single, scalar reward. Most commonly this is a linear weighted sum of the individual objective rewards (e.g. [3, 4]). The choice of weights allows the user some control over the nature of the policy found by the system, by placing greater or lesser emphasis on each objective. Less frequently a more complex, non-linear function tuned to the problem domain may be used [2]. The primary advantage of linear scalarisation lies in its simplicity – it can readily be integrated into existing RL algorithms with very little modification. However linear scalarisation has a fundamental limitation, in that it can not find solutions which lie in concave or linear regions of the Pareto front. [5] demonstrated for a number of benchmark problems that the Pareto fronts contain a substantial number of solutions which can not be found via linear scalarisation.

A small number of alternatives to scalarisation have been investigated. [6] used lexicographic ordering and thresholding of objectives for problems with constraints for certain objectives (e.g. a robot maintaining an energy level greater than zero whilst accomplishing some task). [7] and [8] describe algorithms where the goal of the agent is to achieve long-term average rewards which lie in an externally defined 'target' region in objective space. These algorithms produce non-stationary policies in which the actions taken by the agent at any point in time are determined both by the current state and by the position of the current average reward vector relative to the target set. [9] developed a policy-gradient MORL algorithm. The algorithm starts from a policy derived by applying RL independently to each objective. This policy is then improved by following gradients in the policy space which are non-negative with regards to all objectives. An approximate Pareto front is constructed by performing repeated searches with different weightings of the gradient directions.

3. Multiobjective Tasks and Stochastic Policies

With the exception of [9], the majority of MORL work has so far focused on finding deterministic policies (policies where the same action is always selected in any given state, other than when an exploratory action is chosen during learning).

This is not surprising as most of these techniques are grounded in single-objective RL algorithms, and in the single-objective case there is little reason to consider stochastic policies as "for any MDP, there exists a stationary, deterministic optimal policy" [9, p288]. However this property of stationary, deterministic policies does not hold true when we consider problems with more than one objective, as illustrated by the simple environment in Fig 3. This environment consists of a single state, with two actions. Both actions lead back to the same state, but their associated rewards correspond to different objectives. Clearly there are only two deterministic policies available: always choosing action a_1 which will maximise the reward on objective 1 but minimise the reward for objective 2, or always choosing action a_2 which will maximise the reward on objective 2 but minimise the reward for objective 1. In any task in which we are considering multiple objectives, inherently we are seeking a solution which is trade-off between these objectives, but in this case neither of these deterministic policies offers any degree of compromise between the two objectives. In contrast, consider a stochastic policy which selects between actions a_1 and a_2 with probabilities p_1 and $(1-p_1)$ respectively. Clearly the average reward received by this policy will be $(p_1, 1-p_1)$. By varying the probability with which each action is selected, a range of policies which offer different compromises between the two objectives can be achieved. For more complex problems usually deterministic policies will exist which do in fact offer a compromise between the different objectives. However these policies will represent discrete, possibly widely-spaced, points in objective-space whereas stochastic policies offer a continuous range of trade-offs between the different objectives. Therefore it likely that a policy which better matches the user's preferences will be found if stochastic policies are considered.

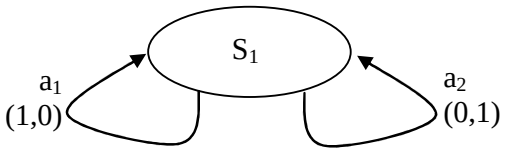

Fig 3: A single-state environment with 2 actions. Performing action a_1 receives a vector reward of (1,0); performing action a_2 receives a vector reward of (0,1).

4. Mixture Policies for Multiobjective Tasks

Having established the potential benefits of stochastic policies for multiobjective tasks, we need to consider how such policies may be found. As noted earlier, many single-objective RL methods (such as the widely used temporal difference approaches such as Q-learning) do not support stochastic policies. Methods such as policy gradient learning and learning automata can learn the probabilities with which each action should be selected in each state of a stochastic policy, and [9] has pioneered the use of policy gradient methods in finding stochastic multiobjective policies.

Rather than considering more sophisticated approaches, this paper will examine means by which simple methods such as Q-learning and scalarisation can be used to find stochastic policies for episodic multiobjective tasks. [9, p59] describes a special

form of stochastic policy known as a mixture policy[2], which is derived from two or more deterministic policies (which we will refer to as base policies). At the start of each episode the mixture policy stochastically selects one of its base policies, which is then followed for the remainder of that episode. Over a large number of episodes, the vector return achieved by this mixture policy will be the mean of that achieved by its base policies, weighted by the probability with which each base policy is selected[3]. In [9], the only base policies considered are those maximising each individual objective, and the mixture policies generated from these base policies are used as the starting point for policy gradient search. Here we consider a more general application of mixture policies in which any set of base policies may be used, and in which the mixture policies themselves are the final outcome of the system.

Consider the generation of mixture policies from a set of deterministic policies. Fig 4 shows the complete set of Pareto optimal deterministic policies for the Deep Sea Treasure task [5]. The line joining policies A and B represents the mixture policies which are derived from that pair of base policies by varying the probability with which the policies are selected. It can be seen that any deterministic policy lying in a concave region of the Pareto front (e.g. policy C) will be dominated by one or more mixture policies derived from policies outside the concave region.

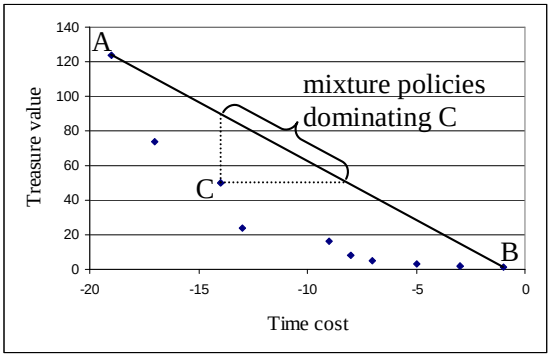

***Fig 4*:** The Deep Sea Treasure task's objective-space. Points are Pareto-optimal deterministic policies. The line *AB* represents mixture policies derived from those base policies.

The Deep Sea Treasure task is unusual in that only the extremal policies are not in a concave portion of the front. Fig 5 illustrates a more generally representative front – the line segments indicate possible mixture policies which can be generated from this front. It can be seen that as the mixture policies are formed via convex combinations[4] of the base policies, the non-dominated mixture policies constructed by this process will form the convex hull of the original base-policy points in objective space [11].

[2] Note that mixture policies are not fully stochastic policies in which actions are chosen stochastically at each state – rather they are a stochastic combination of deterministic policies.

[3] It is important to note that this is only true because the choice between policies *A* and *B* is made at the start of each episode. Switching between policies at other time-steps would likely result in erratic and sub-optimal behaviour.

[4] A convex combination is a linear combination of vectors, in which the weights sum to 1.

All points (deterministic policies) from the original front which are not on this hull will be dominated once the mixture policies are considered. Similarly mixture policies derived from policies which are not neighbouring members of the convex hull will also be dominated by at least one other mixture policy.

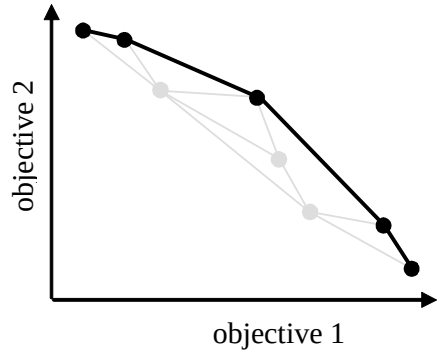

Fig 5: A hypothetical Pareto front. Points indicate Pareto-optimal deterministic policies (black points indicate policies on the convex hull of the front). Lines indicate mixture policies generated from pairs of deterministic base policies (some combinations of base policies have been omitted for reasons of clarity).

In summary mixture policies provide two potential benefits for episodic tasks when compared to deterministic policies. First they provide a continuous range of trade-offs between the objectives as opposed to the discrete set of trade-offs embodied by the deterministic policies. Hence they are likely to provide a more precise match to the preferences of the decision maker. Second for problems where the Pareto set of deterministic policies contains concave regions (which was shown to be the case for all benchmark problems examined by [5]) mixture policies exist which dominate (in some cases by a significant margin) some otherwise non-dominated deterministic policies.

5 Selecting and Constructing Mixture Policies

In light of the benefits outlined in Section 4, we propose the following general approach to finding suitable policies for episodic multiobjective tasks:

- generate a set of Pareto-optimal deterministic policies
- use these policies as base policies to derive a set of mixture policies
- select a mixture policy which is appropriate to the preferences of the decision-maker who is using this MORL system

As evident in Fig 5, a mixture policy will be non-dominated if and only if it is formed from base policies which are neighbouring points on the convex hull of the Pareto front. Therefore the set of base policies need consist only of Pareto-optimal policies which are not in concave regions of the Pareto front – any computation expended in finding policies in concavities is wasted as they will not be used by any non-

dominated mixture policy. Interestingly in this context the inability of scalarised MORL to discover policies other than those on the convex hull (see Section 2) may in fact make it more efficient than other methods which can find such policies. A simple approach to finding the set of base policies is to carry out repeated runs of a scalarised RL algorithm, varying the scalarising weights between runs. A more efficient approach is Convex Hull Value Iteration [12] which finds in parallel all deterministic policies lying on the convex hull without any need for an explicit search through the scalarising weight space.

Once the base policies have been found, their neighbourhood relationships can be established through a range of algorithms (see [13] for a summary). This provides the information required to derive the complete set of Pareto-optimal mixture polices. This set is then displayed to the system's user so they can select the single mixture policy which best fits their preferences. In the following two sub-sections we will describe two approaches to this process of selecting a mixture policy, depending on the number of objectives involved in the task.

5.1 Convex Hull Visualisation and Barycentric Coefficients

For problems with a low dimensionality (two or three objectives) the set of mixture policies can be directly displayed to the user via 2-dimensional, 3-dimensional or stereo graphics. This provides the user with a clear depiction of the relationships between the objectives, and allows them to make an informed choice of the best available policy for their needs. The choice of policy can be indicated by selecting a point anywhere on the surface of the hull. Once the user selects a point in objective-space it is simply a matter of determining the probabilistic weightings of base policies required to construct a mixture policy to achieve that combination of rewards. This can be done by calculating the barycentric coordinates of the target point, and using these as the probability of selection for each base policy. Barycentric coordinates are coordinates defined in terms of the vertices of a simplex (in our case, the points in objective space corresponding to the base policies, which we will label as $V_1,V_2,..,V_n$). If V_0 is the target point (the objective-space position of our desired mixture policy) and $V_{i,j}$ denotes the value of a point V_i for objective j, then the barycentric coordinates are values $b_1..b_n$ such that the following equality holds for all j=1,...,n:

$$V_{0,j} = \sum_{i=1}^{n} b_i V_{i,j}$$

If V_0 lies within the simplex defined by $V_1,...,V_n$ (which is always true in our case due to the manner in which V_0 is specified by the user) then the following property holds, and therefore the barycentric coefficient b_i can be directly interpreted as the probability with which base policy i will be selected at the start of each episode:

$$\sum_{i=1}^{n} b_i = 1$$

For a problem with two objectives, each mixture policy lies on the line-segment bounded by the base policy points V_1 and V_2, as shown in Fig 6. In this case the barycentric coefficients can easily be calculated from the ratios of the line-segments V_1V_0 and V_2V_0:

$$b_i = \frac{\|V_i - V_0\|}{\|V_2 - V_1\|}$$

For a problem with three objectives, V_0 will lie within the bounds of a planar triangle defined by V_1, V_2 and V_3. In this case the value of the coefficient for each vertex can be calculated based on the percentage of the area of this triangle which is occupied by the sub-triangle formed by V_0 and the other two vertices (see Fig 7). The area calculations can be efficiently implemented using vector cross-products and length operations as follows:

$$A_1 = \|(V_2 - V_0) \times (V_3 - V_0)\|$$
$$A_2 = \|(V_1 - V_0) \times (V_3 - V_0)\|$$
$$A_3 = \|(V_1 - V_0) \times (V_2 - V_0)\|$$
$$b_i = \frac{A_i}{\sum_{j=1}^{3} A_j}$$

This barycentric approach to constructing mixture policies can be extended to higher dimensions. However as the number of objectives rises beyond three, direct visualisation of the front becomes problematic[5] and the computational cost of establishing hull geometry and calculating the barycentric coefficients increases. Therefore Section 5.2 discusses an interactive approach which does not require visualisation of the hull.

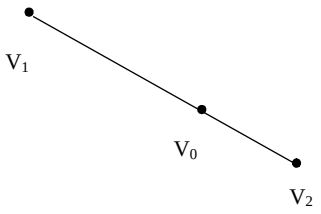

Fig 6: For the two objective case, each mixture policy V_0 lies on the line-segment formed by the base policies V_1 and V_2.

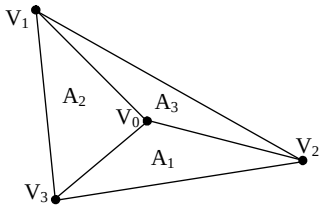

Fig 7: For the three objective case, each mixture policy V_0 lies in the triangle formed by the base policies V_1, V_2 and V_3, and the barycentric coefficients can be calculated from the relative areas of the sub-triangles A_1, A_2 and A_3.

[5] Although visualisation of high-dimensional Pareto fronts has been explored in the multiobjective optimisation community – see for example [14].

5.2 Interactive Construction of a Mixture Policy

Where direct selection of mixture policies from a visualisation of the hull is not practical (i.e. where the number of objectives exceeds three), an alternative approach is to allow the user to directly select the base policies and set their probabilities so as to form a suitable mixture policy. This can be achieved via the following process:

1. A complete list of possible base policies is presented to the user textually, augmented by lower-dimensional visualisations.
2. The user selects the base policy which most closely matches their preferences.
3. If the number of selected policies is less than the number of objectives, remove from the list all policies which are not neighbours on the hull of all selected policies and return to Step 2.
4. An initial mixture policy is constructed from an equal weighting of the selected base policies. The user manipulates the base policy probabilities using a linked set of sliders (as one probability is adjusted, the other sliders are adjusted in the opposite direction) whilst the reward vector for the current mixture policy is displayed. The user can explore the trade-offs available based on the currently selected base policies, before settling on a mixture policy, or returning to Step 1 to select a new set of base policies.

6 Conclusions and Future Work

This paper has demonstrated the utility of stochastic policies for multiobjective tasks. Stochastic policies offer a continuous range of solutions, as opposed to the discrete set of solutions offered by deterministic policies which may contain large gaps between neighbouring solutions. Hence it is more likely that a policy closely matching the user's preferences will be discovered if stochastic policies are allowed. In addition it has been shown that for some problems stochastic policies can be superior (in the sense of Pareto dominance) to the best deterministic policies. We have shown that for episodic tasks, mixture policies offer an inexpensive means of gaining the benefits afforded by stochasticity. The base deterministic policies required to construct mixture policies can be found efficiently using Convex Hull Value Iteration, and mixture policies can then be derived with relatively little computational cost and no further interaction with the environment. Two approaches for constructing a suitable mixture policy have been proposed for problems with different numbers of objectives.

It is important to note one fundamental limitation of the approaches described in this paper – mixture policies can only be applied in this manner to tasks which are known to be episodic. The start of a new episode is used as a trigger for stochastically selecting a base policy to follow – switching between base policies at any other time would likely lead to erratic, sub-optimal performance by the agent, such as oscillating between two locations. The benefits of stochastic policies still apply to non-episodic tasks however, and so an important direction for future research is to examine whether suitable switching states can be identified for mixture policies for such tasks. A second possible limitation is that for some tasks consistency of behaviour may itself be a desirable feature, and therefore stochastic policies may be unacceptable. To

handle these situations there is still a need for MORL systems which can identify all Pareto-optimal deterministic policies, not just those on the convex hull.

Acknowledgements

We wish to acknowledge the insight into the relationship between Pareto fronts and online learning provided by Dr Peter Andreae from Victoria University of Wellington, and the useful feedback provided by this paper's referees.

References

1. Coello Coello, C.A.: Handling Preferences in Evolutionary Multiobjective Optimization: A Survey, in 2000 Congress on Evolutionary Computation, Vol 1, 30--37 (2000)
2. Tesauro, G., Das, R., Chan, H., Kephart, J. O., Lefurgy, C., Levine, D. W., and Rawson, F.: Managing power consumption and performance of computing systems using reinforcement learning. Neural Information Processing Systems (2007)
3. Natarajan, S., & Tadepalli, P.: Dynamic preferences in multi-criteria reinforcement learning, International Conference on Machine Learning, Bonn, Germany, 601--608 (2005).
4. Castelletti, A., Corani, G., Rizzolli, A., Soncinie-Sessa, R., & Weber, E.: Reinforcement learning in the operational management of a water system. IFAC Workshop on Modeling and Control in Environmental Issues, Keio University, Yokohama, Japan. 325--330 (2002)
5. Vamplew, P., Yearwood, J., Dazeley, R. and Berry, A., On the Limitations of Scalarisation for Multiobjective Learning of Pareto Fronts, in Wobcke, W and Zhang, M (Eds), AI08: The 21st Australasian Conference on Artificial Intelligence, Auckland, New Zealand, December 2008, Springer, Lecture Notes in AI, Vol. 5360, pp 372-378 (2008)
6. Gabor, Z., Kalmar, Z., & Szepesvari, C.: Multi-criteria reinforcement learning. The Fifteenth International Conference on Machine Learning, 197--205 (1998)
7. Mannor, S., & Shimkin, N.: The steering approach for multi-criteria reinforcement learning. Neural Information Processing Systems, Vancouver, Canada. 1563—1570 (2001)
8. Mannor, S., & Shimkin, N.: A geometric approach to multi-criterion reinforcement learning. Journal of Machine Learning Research, 5, 325—360 (2004)
9. Shelton, C. R.: Importance sampling for reinforcement learning with multiple objectives, Massachusetts Institute of Technology AI Laboratory Tech Report No. 2001-003 (2001)
10. Mahadevan, S., Ghavamzadeh, M., Theocharous, G. and Rohanimanesh, K.: Hierarchical Approaches to Concurrency, Multiagency, and Partial Observability, in Si, J, Barto, A, Powell, W and Wunsch, D (eds) "Handbook of Learning and Adaptive Dynamic Programming", Wiley-IEEE, 285—310 (2004)
11. Kelley, J.L. & Namioka, I. Linear topological spaces, Graduate Texts in Mathematics, No. 36. Springer-Verlag, New York-Heidelberg (1976)
12. Barrett, L. and Narayanan, S., Learning All Optimal Policies with Multiple Criteria, Proceedings of the International Conference on Machine Learning (2008).
13. Seidel, R., Convex Hull Computations, in Goodman, J.E. and O'Rourke, J. (eds.), Handbook of Discrete and Computational Geometry, CRC Press, pp 361—376 (1997)
14. Agrawal, G., Lewis, K., Chugh, K., Huang, C.-H., Parashar, S. and Bloebaum, C. L., Intuitive Visualization of Pareto Frontier for Multi-Objective Optimization in n-Dimensional Performance Space, 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, NY (2004)